My Observation on Common Pattern of Linear Time Algorithms

Tri Nguyen

nguyetr9@oregonstate.edu

August 16, 2023

As preparing for coding interview, I have been practicing solving coding problems. It was struggling to me many times to come up with solution within reasonable time. Then, I think there should be a better way to deal with these problems once and for all. So I came up with this unified view on how to solve a particular class of problems, i.e., problems of finding some value and they have optimal algorithms with O(n) running time and O(1) memory complexity.

Warning: This view has never been tested in real interview, so viewer discretion is advised. ;))

1 General Strategy

Using recursion strategy to solve this problem: Suppose we have solution for nums[0..k], how to produce solution for nums[0..k+1]? Or more generally, imagine that we have a "state[k]" assocciating to the input nums[0..k], then

- how to deduce desired solution from "state[k]"
- how to deduce the next "state[k+1]" associcating to input nums[0..k+1] based on the previous "state[k]" and the new element nums[k+1]

The requirement on state.

- In term of memory, total memory to save state of step should be O(1). The only way to realize this is to have a fixed amount of memory for "state".
- In term of computation, total computation cost of all step should be O(n). A trivial way is to compute O(1) for each step. Beside these trivial realization, we can distribute memory/computation unevenly among all steps, as long as the summation is O(n).
- I am not sure if there are other ways.

2 Examples

2.1 Find the maximum value

2.2 The daily temperature problem

2.3 The maximum subarray problem